

# SpInside Macintosh<sup>TM</sup>

An on-line development prototype of  
Apple's *Inside Macintosh* Volumes I–V

|                                    |
|------------------------------------|
| Stack Version: 1.0 • November 1989 |
| HyperCard Version: 2.4.1           |
| System Version: 8.96               |
| Monitors Setting: 8-bit, Color     |



Apple  
Developer Group

Copyright © 1985-1989 by Apple Computer, Inc.

---

## THE DEFERRED TASK MANAGER

---

About This Chapter  
About the Deferred Task Manager  
Deferred Task Manager Routine  
Summary of the Deferred Task Manager

---

## ABOUT THIS CHAPTER

---

This chapter describes the Deferred Task Manager, which provides improved interrupt handling by allowing lengthy tasks to be deferred.

Reader's guide: Lengthy tasks are usually initiated by slot cards. Hence you normally need the information in this chapter only if your program deals with slot card interrupts.

---

## ABOUT THE DEFERRED TASK MANAGER

---

The Deferred Task Manager provides a way to defer the execution of interrupt tasks until interrupts have been reenabled (processor priority level 0). It maintains a deferred task queue; instead of performing a task immediately, you can place the information describing the task into the queue by calling the DTInstall procedure. All system interrupt handlers check this queue just before returning. If there are tasks in the queue and interrupts are about to be reenabled, the tasks are removed and then executed with all interrupts enabled.

While useful for all types of interrupt tasks, the Deferred Task Manager is especially handy for slot interrupts. Interrupts from NuBus slot devices are received and decoded by the VIA2, a second Versatile Interface Adapter (Rockwell 6522) chip on the Macintosh II. The VIA2 generates level-2 interrupts and, due to the way the VIA chip works, interrupts must be serviced before the processor priority level can be lowered (otherwise, a system error will occur). During this period (which could be quite long depending on the slot device) other level-2 interrupts such as those for sound, as well as all level-1 interrupts, are blocked. By using the Deferred Task Manager, the processing of slot interrupts can be deferred until all the slots are scanned; just before returning, the slot interrupt handler dispatches to any tasks in the deferred task queue.

The deferred task queue is a standard Macintosh Operating System queue, as described in the Operating System Utilities chapter. Each entry in the deferred task queue has the following structure:

```
TYPE DeferredTask = RECORD
    qLink:      QElemPtr;  {next queue entry}
    qType:      INTEGER;   {queue type}
    dtFlags:    INTEGER;   {reserved}
    dtAddr:     ProcPtr;   {pointer to task}
    dtParm:     LONGINT;   {optional parameter}
    dtReserved: LONGINT    {reserved--should be 0}
END;
```

QLink points to the next entry in the queue, and qType indicates the queue type, which must always be ORD(dtQType).

DTAddr contains a pointer to the task. DTParm is useful only from assembly language.

Assembly-language note: DTParm lets you pass an optional parameter to be loaded into register A1 just before the task is executed.

---

## DEFERRED TASK MANAGER ROUTINES

---

FUNCTION DTInstall (dtTaskPtr: QElemPtr) : OSErr;

Trap macro \_DTInstall

On entry A0: dtTaskPtr (pointer)

On exit D0: result code (word)

Note: To reduce overhead at interrupt time, instead of executing the \_DTInstall trap you can load the jump vector jDTInstall into an address register other than A0 and execute a JSR instruction using that register.

DTInstall adds the specified task to the deferred task queue. Your application must fill in all fields of the task except qLink. DTInstall returns one of the result codes listed below.

|              |         |                       |
|--------------|---------|-----------------------|
| Result codes | noErr   | No error              |
|              | vTypErr | Invalid queue element |

---

## SUMMARY OF THE DEFERRED TASK MANAGER

---

### Data Types

#### TYPE

```
DeferredTask = RECORD
    qLink:      QElemPtr;  {next queue entry}
    qType:      INTEGER;   {queue type}
    dtFlags:    INTEGER;   {reserved}
    dtAddr:     ProcPtr;   {pointer to task}
    dtParm:     LONGINT;   {optional parameter}
    dtReserved: LONGINT    {reserved--should be 0}
END;
```

---

### Routines

```
FUNCTION DTInstall (dtTaskPtr: QElemPtr) : OSErr;
```

---

### Assembly-Language Information

#### Routines

| Trap macro | On entry            | On exit                |
|------------|---------------------|------------------------|
| _DTInstall | A0: dtTaskPtr (ptr) | D0: result code (word) |

#### Structure of Deferred Task Manager Queue Entry

|          |                                |
|----------|--------------------------------|
| qLink    | Pointer to next queue entry    |
| qType    | Queue type (word)              |
| dtFlags  | Reserved (word)                |
| dtAddr   | Address of task                |
| dtParm   | Optional parameter (long)      |
| dtResrvd | Reserved--should be 0 (long)   |
| dtQESize | Size in bytes of queue element |

#### Variables

|            |                                       |
|------------|---------------------------------------|
| DTQueue    | Deferred task queue header (10 bytes) |
| JDTInstall | Jump vector for DTInstall routine     |



END OF DOCUMENT